REMARKS/ARGUMENTS

Favorable reconsideration of this application is respectfully requested.

Claims 1, 3, 7, 8, 9, 11, 15, 16, 18, 19, and 21-24 are pending in this application. Claims 2, 4, 5, 6, 10, 12, 13, 14, 17, and 20 are canceled by the present response without prejudice.

Claims 21-24 were rejected under 35 U.S.C. § 112, first paragraph. Claims 1-18 and 21-24 were rejected under 35 U.S.C. § 112, second paragraph. Claims 1, 9, 16, and 19 were rejected under 35 U.S.C. § 103(a) as unpatentable over U.S. patent 6,182,281 to Nackman et al. (herein "Nackman") in view of U.S. patent 5,613,120 to Palay et al. (herein "Palay") and further in view of U.S. patent 5,175,856 to Van Dyke et al. (herein "Van Dyke") and U.S. patent 6,446,254 to Chapman et al. (herein "Chapman"). Claims 2, 3, 10, 11, 17, 18, and 20 were rejected under 35 U.S.C. § 103(a) as unpatentable over Nackman in view of Palay and further in view of Van Dyke, Chapman and "Access 2000 for Windows for Dummies" by John Kaufeld, 1999 (herein "Kaufeld"). Claims 4 and 12 were rejected under 35 U.S.C. § 103(a) as unpatentable over Nackman in view of Palay and further in view of Van Dyke, Chapman, Kaufeld and U.S. patent 6,041,179 to Bacon et al. (herein "Bacon").

Addressing now the rejection of claims 21-24 under 35 U.S.C. § 112, first paragraph, that rejection is traversed by the present response.

Claims 21-24 were rejected as the described "second coat optimizing processor ..." was not supported by the original specification and contradicted the disclosure in the specification at page 25, lines 21-23.

In response to that position each of claims 21-24 is amended by the present response to no longer recite a "second coat optimizing processor ...". Further, the claims are amended to clarify features from the original specification, and to recite instantaneous information

referred from the multiphase data type definition table for deciding generation of each instance of the declared multiphase data type definition.

Such claim amendments are fully supported by the original specification, for example at page 16, line 23 to page 20, line 25, and see also Figures 14-16A, 16B in the present specification.

The presently submitted amendments to claims 21-24 are believed to address the rejections of those claims under 35 U.S.C. § 112, first paragraph.

Addressing now the rejection of claims 1-18 and 21-24 under 35 U.S.C. § 112, second paragraph, that rejection is traversed by the present response.

Each of the above-noted claims is amended by the present response to clarify features recited therein. The above-noted claims were rejected as the term "data type definition" was unclear. The claims are amended to now clarify that subject matter. More particularly, claims 1, 9, 16, and 21-24 are amended to clarify the distinction between data type definition described in a source program to be compiled and a name of a data type definition registered in a data type definition table. Claims 1, 9, 16, and 21-24 also clarify the operation before deletion of the data type definition from the source program.

Further, claims 5, 6, 13, and 14 are canceled by the present response and the other claims are amended by the present response to clarify the term "multiphase".

Further, with respect to the rejection to claims 5 and 13 as reciting the phrase "is one of" and to claims 6 and 14 as reciting the phrase "representing whether instantiation is requested in the source program for every symbol of each data type of the multiphase type", those phrases were clarified or deleted in the previous amendment, and thus the rejection thereof is unclear. However, the pending claims are amended to even further clarify such subject matter.

Again the presently submitted claim amendments are believed to be fully supported by the original specification, see for example page 16, line 23 to page 20, line 25 and Figures 14-16A, 16B.

Addressing now the rejection of claims 1, 9, 16, and 19 under 35 U.S.C. § 103(a) as unpatentable over Nackman in view of Palay and further in view of Van Dyke and Chapman, that rejection is traversed by the present response.

Each of the currently pending claims is amended by the present response to clarify features recited therein. Specifically, each of the independent claims clarifies that the data type definition table, which is arranged for one object program, stores "a set of a name of data type definition for data or a function in the source program, and a use flag of the name".

The claims also clarify the code optimizing processing, and in particularly claim 1 now clarifies features of:

> a data type definition detector configured to detect a predetermined data type definition declared in the preprocessed source program;
>
> a first table updating module configured to, if a name of the detected data type definition is not registered, register the name of the detected data type definition into the data type definition table;
>
> a first source updating module configured to, if the name of the detected data type definition is registered, delete the data type definition in the source program;
>
> a second table updating module configured to, if the data type definition is described in a body of any of the source programs to be linked into the one object program, set the use flag to a use status;
>
> a second source updating module configured to delete the data type definition of which the use flag is set to the use status from all the source programs to optimize the source programs; ...

The other independent claims 9, 16, 19, and 21-24 are similarly amended as in independent claim 1 noted above.

In the outstanding Office Action Nackman is cited to teach "(a) a preprocessor configured to execute preprocessing of source programs (Column 18, lines 44-50); and (b) a language processor configured to compile the source programs (Column 18, lines 3-13)".[1] The outstanding Office Action further relies upon Palay to teach a "class definition table (Column 12, lines 15-17) and a linker that removes data type definitions from the object file when the definition is already stored in the class definition table (Column 28, lines 39-61)".[2] Van Dyke is relied upon to "teach a use flag that specifies when a data type has been explicitly declared (Column 25, Table VIII)",[3] and Chapman is relied upon to teach "removing unused classes from a source file (Column 4, lines 27-43)".[4]

However, applicants traverse the above-noted rejections as the noted teachings do not fully meet the features now clarified in the pending claims.

More specifically, no combination of teachings of Nackman, Palay, Van Dyke, and Chapman fully meets the clarified features in the claims of code optimizing operations in which a first table updating module registers, if a name of the detected data type definition is not registered, the name of the detected data type definition into the data type definition table; a first source updating module deletes, if the name of the detected data type definition is registered, the data type definition of the source program; a second table updating module sets, if the data type definition is described in the body of any of the source programs to be linked into the one object program, the use flag to a use status; and a second source updating module deletes the data type definition of which the use flag is set to the use status from all the source programs to optimize the source programs.

Moreover, Chapman only relates to an interpreted language environment such as JAVA and Smalltalk in which there is no request for compiling of source programs, and

[1] Office Action of August 26, 2003, page 4, lines 5-7 of prenumbered paragraph 6.
[2] Office Action of August 26, 2003, page 4, lines 9-11 of prenumbered paragraph 6.
[3] Office Action of August 26, 2003, page 5, lines 3-4.
[4] Office Action of August 26, 2003, page 5, lines 6-7.

where only ROM memory utilization is optimized. In contrast to Chapman, Nackman and Palay are directed to compiling language environments. As a result, the teachings in Chapman are not properly combinable to the teachings in Nackman and Palay as they are directed to different environments with different objectives.

Moreover, Palay merely discloses a class definition table 414 containing a list of class definition (see column 12, lines 15-17) and a linker that removes duplicate class definitions between a plurality of object files 106, 108 (see column 28, lines 39-61). In contrast to Palay, in the claims as currently written the code optimizing process can eliminate unnecessary data type definitions twice in one object file.

In such ways, the claims as currently written are believed to clearly distinguish over the combination of teachings in Nackman, Palay, Van Dyke, and Chapman.

Moreover, in the further rejections based on Kaufeld and Bacon, no teachings in Kaufeld or Bacon can overcome the above-noted deficiencies of Nackman, Palay, Van Dyke, and Chapman, and thus those further rejections are also overcome by the present response.

As no other issues are pending in this application, it is respectfully submitted that the present application is now in condition for allowance, and it is hereby respectfully requested that this case be passed to issue.

Respectfully submitted,

OBLON, SPIVAK, McCLELLAND,
MAIER & NEUSTADT, P.C.

Eckhard H. Kuesters
Registration No. 28,870
Surinder Sachar
Registration No. 34,423
Attorneys of Record

Customer Number
22850

Tel: (703) 413-3000
Fax: (703) 413-2220

EHK:SNS\la
I:\ATTY\SNS\19's\193857\193857US-AF.DOC